Alario-Hoyos, C., Estévez-Ayres, I., Gallego-Romero, J.M., Delgado Kloos, C., Fernández-Panadero, C., Crespo-García, R.M., Almenares, F., ... Blasco, J. (2018). A Study of Learning-by-Doing in MOOCs through the Integration of Third-Party External Tools: Comparison of Synchronous and Asynchronous Running Modes. *Journal of Universal Computer Science*, 24(8), (2018), pp. 1015-1033.

DOI: https://doi.org/10.3217/jucs-024-08-1015

Journal of Universal Computer Science, vol. 24, no. 8 (2018), 1015-1033
submitted: 14/12/17, accepted: 5/7/18, appeared: 28/8/18 © J.UCS

A Study of Learning-by-Doing in MOOCs through the Integration of Third-Party External Tools: Comparison of Synchronous and Asynchronous Running Modes

Carlos Alario-Hoyos, Iria Estévez-Ayres, Jesús M. Gallego-Romero
Carlos Delgado Kloos, Carmen Fernández-Panadero, Raquel M. Crespo-García
Florina Almenares, María Blanca Ibáñez, Julio Villena-Román
Jorge Ruiz-Magaña
(Universidad Carlos III de Madrid, Spain
{calario@it, ayres@it, jegalleg@pa, cdk@it, mcfp@it, rcrespo@it, florina@it, mbibanez@it
jvillena@it, jrmagana@it}.uc3m.es)

Jorge Blasco
(Royal Holloway, University of London, UK
Jorge.BlascoAlis@rhul.ac.uk)

Abstract: Many MOOCs are being designed replicating traditional passive teaching approaches but using video lectures as the means of transmitting information. However, it is well known that learning-by-doing increases retention rates and, thus, allows achieving a more effective learning. To this end, it is worth exploring which tools fit best in the context of each MOOC to enrich learners' experience, including built-in tools already available in the MOOC platform, and third-party external tools which can be integrated in the MOOC platform. This paper presents an example of the integration of a software development tool, called Codeboard, in three MOOCs which serve as an introduction to programming with Java. We analyze the effect this tool has on learners' interaction and engagement when running the MOOCs in synchronous (instructor-paced) or asynchronous (self-paced) modes. Results show that the overall use of the tool is similar, regardless of the course running mode, although in the case of the synchronous mode the use of the tool is concentrated in a shorter period of time. Results also show that in the synchronous mode there is a higher percentage of accesses to the tool from registered learners (who can save their advances and continue the work later); this finding suggests that learners in the synchronous running mode are more engaged with the MOOC.

Keywords: MOOCs, programming, tools, Codeboard, instructor-paced, self-paced
Categories: D.2.3, D.2.12, K.3.1

1 Introduction

Most Massive Open Online Courses (MOOCs) offered in major platforms, such as Coursera or edX, share a set of common features [Bali, 2014], making intensive use of video lectures complemented by formative and summative activities, additional resources (e.g., readings, animations...), and discussion forums. However, teachers should bear in mind the need for learners to practice and apply the concepts explained in order to increase learning retention rates and achieve a more effective learning [Clarke and Braun, 2013]. To foster learning-by-doing in MOOCs, teachers can take advantage of the built-in tools provided by MOOC platforms (e.g., automatic

correction quizzes and peer-assessment activities), considering also the possibility of integrating third-party external tools to reinforce the application of theoretical concepts to solve problems or work in small projects [Cruz-Benito et al., 2015].

There are two main modes to run a MOOC [Campbell et al., 2014]. The first one is the synchronous (also called instructor-paced, or instructor-led) mode. In this mode, course materials are released on a weekly basis and, generally, there are regular deadlines for assignments; this mode allows higher interaction among learners, as most of them start and finish the course at the same time, and also facilitates the possibility of carrying out activities that rely on massiveness, such as peer evaluation [Suen, 2014], but can increase dropout rates by having tight deadlines, particularly affecting latecomers who cannot catch up. In general, learners' interaction with educational resources in synchronous MOOCs is concentrated in the first weeks, showing an exponential decrease in the level of daily activity from the beginning to the end of the course [Alario-Hoyos et al., 2014]. The second running mode is the asynchronous (also called self-paced, or learner-paced) mode. In this mode, all materials are released from the beginning, and the course is open indefinitely or for long periods of time; this mode allows more flexibility for learners to complete the course without the pressure of regular deadlines [Rizzuto, 2017], but can lead to less interaction among learners, as they might follow different paces. This delivery mode is more preferred by learners when they see the topic of the MOOC as an entertainment [Watson et al. 2017], or when there is some level of tailoring of the MOOC content to learners' preferences [Crosslin, 2018]. Unlike in the previous mode, MOOCs offered in asynchronous mode maintain a homogeneous level of activity after the first few weeks, as the interactions of new enrollees make up for those of learners who drop out [Campbell et al., 2014].

This paper researches the effect on learners' interaction and engagement of third-party tool integration and MOOC running modes. The integrated tool used as example is Codeboard (codeboard.io), a web-based development environment which allows compiling and running Java code directly from the browser; this way the learner sees the activities to be done with Codeboard integrated as part of the learning sequences of the MOOC, which is intended to cause a feeling of seamlessness, as everything is done in the same tab of the web browser. Codeboard is integrated in three MOOCs on "Introduction to Programming with Java," deployed in edX, and offered in synchronous and asynchronous running modes throughout three years, reaching more than 300,000 enrollees overall.

In this context, the following research questions (RQ) arise:

- (RQ1) Is Codeboard a useful tool to promote learning-by-doing in MOOCs?
- (RQ2) Do learners' interaction with Codeboard differ when the MOOC is offered in synchronous and asynchronous running modes?
- (RQ3) Do learners' engagement with Codeboard differ when the MOOC is offered in synchronous and asynchronous running modes?

The rest of the paper is structured as follows. Section 2 analyzes the literature in relation to interaction in MOOCs. Section 3 presents the methods used in the study, describing the MOOCs used for the analysis, the type and number of activities integrated in Codeboard, and the data collected. Section 4 presents and discusses the results that allow answering the abovementioned research questions. Limitations of the study are presented in Section 5, with conclusions in Section 6.

2 Related work

It is widely accepted that one of the best ways of reinforcing the theoretical concepts explained by a teacher is by applying them. Although learning-by-doing has consolidated over the years in face-to-face education [Felder and Brent, 2003], it still poses significant challenges for distance education. Anido et al. [Anido et al., 2001] classified the main approaches to learning-by-doing in digital environments in: (a) accessing to real equipment through an Internet interface (e.g., virtual/remote labs), and (b) using simulators. Thus, learning-by-doing in online education entails necessarily interaction with additional resources and tools, which in many cases are external to the platforms where the main educational contents are deployed.

In the case of MOOCs, this is particularly challenging, since there might be thousands of learners interacting at the same time with the resources or tools intended to promote learning-by-doing. In fact, practical activities which promote learning-by-doing are seen as one of the keys to engagement of MOOC learners [Hew, 2016], with the lack of these practical activities identified as one of the main reasons of failures in MOOCs from the learner's perspective [Rai and Chun-Rao, 2016]. The relevance of selecting appropriate resources and tools and how these are accessed by learners is noticeable due to the correlation found between them and learners' success [Breslow et al., 2013; Champaign et al., 2014]. Therefore, it is important to design MOOCs that include active learning experiences which use several resources and tools in order to foster learners' engagement [Delgado-Kloos et al., 2017].

Although MOOC platforms typically include a variety of built-in tools for instructors to add to their courses, most of these tools can only be used for creating quizzes or peer-review activities [Admiraal et al., 2015]. If the teacher wants to offer other kinds of activities, then there are mainly two choices: (1) implementing an ad-hoc tool for that MOOC platform (provided that the platform is open source or includes programming interfaces that allow extending it); or (2) integrating external tools (provided that the platform supports some kind of third-party integration) [Cruz-Benito et al., 2015; Staubitz et al., 2014]. The first option is less portable; for example, if the instructors want to reuse their courses in another platform, a new development for such platform is needed. The second option, although easier for the creation and reuse of content, entails to rely not only on the specific MOOC platform, but also on the services provided by third-parties [Alario-Hoyos and Wilson, 2010].

Regarding the second option, there are protocols and standards that facilitate the communication between the platform and the external tool. For example, the well-known IMS LTI standard [IMS, 2012] allows a loosely-coupled integration between external tools and platforms, including traditional LMSs (e.g., Moodle, Blackboard) and MOOC platforms (e.g., edX, Coursera). The literature reports a frequent use of the IMS LTI standard to integrate external tools in traditional LMSs [Fontela et al., 2011; Forment et al., 2012; Queirós et al., 2016]. In the case of MOOCs, however, there is little work reporting on success cases of external tools integration through IMS LTI (or through other communication protocols). Exceptions are the works by Aleven et al. [Aleven et al, 2016], who integrated Intelligent Tutoring Systems in edX, by Bhatnagar et al. [Bhatnagar et al., 2016], who integrated an asynchronous peer instruction component as part of three MOOCs on edX, or by Staubitz and Meinel [Staubitz and Meinel, 2017], who integrated a toolset for collaboration and

teamwork in their MOOC platform. One particularly interesting case is the one of Codeboard, a web-based development environment which supports IMS LTI, and which has been integrated in several MOOCs already to teach learners to code through learning-by-doing [Morales Chan et al., 2017; Krugel and Hubwieser, 2017].

The integration of Codeboard to promote learning-by-doing in six runs of three different MOOCs is precisely the focus of this research, with the novelty of comparing two running modes: synchronous and asynchronous. Few works in the literature tried to shed some light on which the best delivery mode for a MOOC from the perspective of learners' interaction and engagement is. Only Campbell et al. [Campbell et al., 2014] studied how the delivery mode can affect learners' behavior. However, this work while providing relevant information for an initial analysis presented significant limitations. First, the same learners who were enrolled in the synchronous mode had access to the archived (asynchronous) course, so these were not independent samples of learners. In addition, the authors did not report on the time the MOOCs were available as archived courses. Finally, the authors only analyzed learners' behavior with videos and quizzes, but not with external tools.

3 Methods

This section first introduces the three MOOCs used in this study, indicating the delivery mode in their different runs. Then, there is a brief description of Codeboard, and its integration in the learning sequences of the MOOCs. Finally, we describe the data which was collected from learners' opinions and interactions with the tool.

3.1 MOOCs on Introduction to Programming with Java

Universidad Carlos III de Madrid (UC3M), Spain, began in 2015 the development of a MOOCs trilogy on "Introduction to Programming with Java," with the aim to teach the world how to start coding from scratch using Java as the driving language. These three MOOCs were offered through edX, initially only in English language. These MOOCs were independent in terms of syllabuses and certifications, although instructors advised learners to have done the first MOOC before enrolling the second one, and the first two MOOCs before enrolling the third one.

The three MOOCs shared similar characteristics: a duration of five weeks each, and an initially estimated learners' workload of between five and seven hours per week. Videos were an important part of these MOOCs, with between 45 and 65 short videos in each MOOC. However, the three MOOCs were designed with the aim to foster learning-by-doing, as they included numerous hands-on activities that make use of both built-in edX tools and third-party external tools [Alario-Hoyos et al., 2016]. The structure of the three MOOCs was similar. Each week included four learning sequences that formed the core contents, with videos interlaced with activities. In addition, every week included a practical laboratory assignment, which had a common storyline per MOOC, and that was intended to build small projects incrementally. There was also a recap sequence per week, with a summary of the key concepts and solutions to the most complex activities, and weekly summative evaluations. Next, there is an overview of each of these three MOOCs:

- "Starting to Code with Java" (**MOOC1**). This MOOC introduced the basics of Java programming with a bottom-up approach, going from imperative programming to object-oriented programming. This MOOC was offered three times. The first run was offered in a *synchronous mode* between April 2015 and June 2015 and, according to edX, had 93,574 enrollees (1,522 completers) (**MOOC1-1R-S**). The second run was offered in an *asynchronous mode* between November 2015 and June 2016 and had 135,468 enrollees (2,027 completers) (**MOOC1-2R-A**). The third run was also offered in an *asynchronous* mode between October 2016 and June 2017 and had 66,788 enrollees (881 completers) (**MOOC1-3R-A**).
- "Writing Good Code" (**MOOC2**). This MOOC introduced mechanisms to improve the quality of code including error detection and correction, testing, complexity, as well as the basis for software engineering and ethical programming. This MOOC was offered in twice. The first run was offered in a *synchronous mode* between April 2016 and June 2016 and had 13,630 enrollees (262 completers) (**MOOC2-1R-S**). The second run was offered in an *asynchronous mode* between November 2016 and June 2017 and had 14,997 enrollees (184 completers) (**MOOC2-2R-A**).
- "Fundamental Data Structures and Algorithm" (**MOOC**). This MOOC introduced basic data structures, such as lists, stacks, queues or trees, and algorithms that can be applied on them, such as inserting, deleting, searching or sorting. This MOOC has been offered once between April 2017 and June 2017, in a *synchronous mode*, getting 8,692 enrollees (90 completers) (**MOOC3-1R-A**).

In total, the three MOOCs were offered three times synchronously (the first time they were all released), and three times asynchronously (subsequent runs), and achieved 333,149 registered learners in total, and 4,966 completers (1.5% of enrollees).

3.2 Integration of Codeboard in the MOOCs

Codeboard is a web-based development environment developed at ETH Zurich, Switzerland, by Christian Estler and Martin Nordio. It is an open source project, that has also a free-to-use option through its website (codeboard.io). Codeboard is very helpful to teach coding in several programming languages, including Java, directly from the browser and without having to install anything locally. For this reason, Codeboard was chosen in the abovementioned MOOCs, as learners could work in small projects without the need to install Java or heavy development environments.

The integration between edX and Codeboard was done using the IMS LTI standard [IMS, 2012]. Instructors created the structure of a Java program directly in Codeboard. This structure included all the Java classes to be used with missing code (attributes, methods, etc.). Then, instructors copied the URLs of each Codeboard activity in the MOOC as part of a learning sequence. Learners could see Codeboard activities integrated in edX as shown in Figure 1. The top bar represents a learning sequence in edX; the learner is now in the last unit (activity) of the learning sequence, which is a practical activity that makes use of Codeboard. The text on top provides instructions to complete that activity. Below, the learner finds Codeboard with a default Java code provided by the teachers. Each learner could independently modify the initial code, compile, and run it (by default, the Java programs provided by the

instructors compiled correctly, although they had missing code). If the page in the browser is reloaded, then the Codeboard activity returns to its initial state. Nevertheless, learners could save their progress if they were registered in Codeboard. This registration process was independent of the registration process in the MOOC and allowed resuming the work in Codeboard activities later.

A total of 117 Codeboard activities of different difficulty levels were created for the three MOOCs. These include those created for the core learning sequences and for the laboratory of each week. In the case of MOOC1, 38 activities were created using Codeboard (19 for the core learning sequences and 19 for the laboratory). In the case of MOOC2, 25 activities were created using Codeboard (19 for the core learning sequences and 6 for the laboratory). In the case of MOOC3, 54 Codeboard activities were created using Codeboard (49 for the core learning sequences and 5 for the laboratory). Each Codeboard activity represented an independent problem in the core learning sequences (although a few of them were improvements over previous ones). In the laboratories, however, Codeboard activities tended to form a project with a common storyline. All Codeboard activities were formative, and therefore did not count towards the calculation of the learner's final grade in the MOOC.

3.3 Data sources

Two main data sources are taken into account to answer the three research questions of this study: (1) a self-reported survey completed by learners, and (2) information provided by Codeboard on its use by learners.

Regarding the first data source, participants in the MOOCs were asked, at the end of the course, to complete an anonymous, voluntary survey, which included, among others, questions on the usefulness of Codeboard and on the degree of difficulty of the proposed Codeboard activities. The information collected from this first data source has a twofold purpose. On the one hand, it serves to answer RQ1 on the usefulness of Codeboard in promoting learning-by-doing in MOOCs. On the other hand, it serves to check whether the difficulty level of the Codeboard activities represented an impediment for the use of the tool in the MOOCs, thus disrupting learners' interaction and engagement (RQ2 and RQ3).

Regarding the second data source, Codeboard collected information about its use for each particular activity. Two types of Codeboard activities were considered: (1) activities that the learner had to solve programming the missing code, and (2) reference solutions to the former activities which were provided at the end of the week, and that could be useful for learners to see how an actual correct solution works. Unfortunately, as Codeboard is a third-party external tool, it was not possible to access low-level data on the use of this tool by MOOC learners, which would have enriched this analysis. Only high-level data displayed through Codeboard graphical user interface could be collected to answer RQ2 and RQ3. This high-level data includes:

- Number of accesses to each Codeboard activity and distribution over time;
- Number of compilations for each Codeboard activity and distribution over time;
- Number of runs for each Codeboard activity and distribution over time;
- Number of accesses to each Codeboard activity by registered users and by anonymous users.

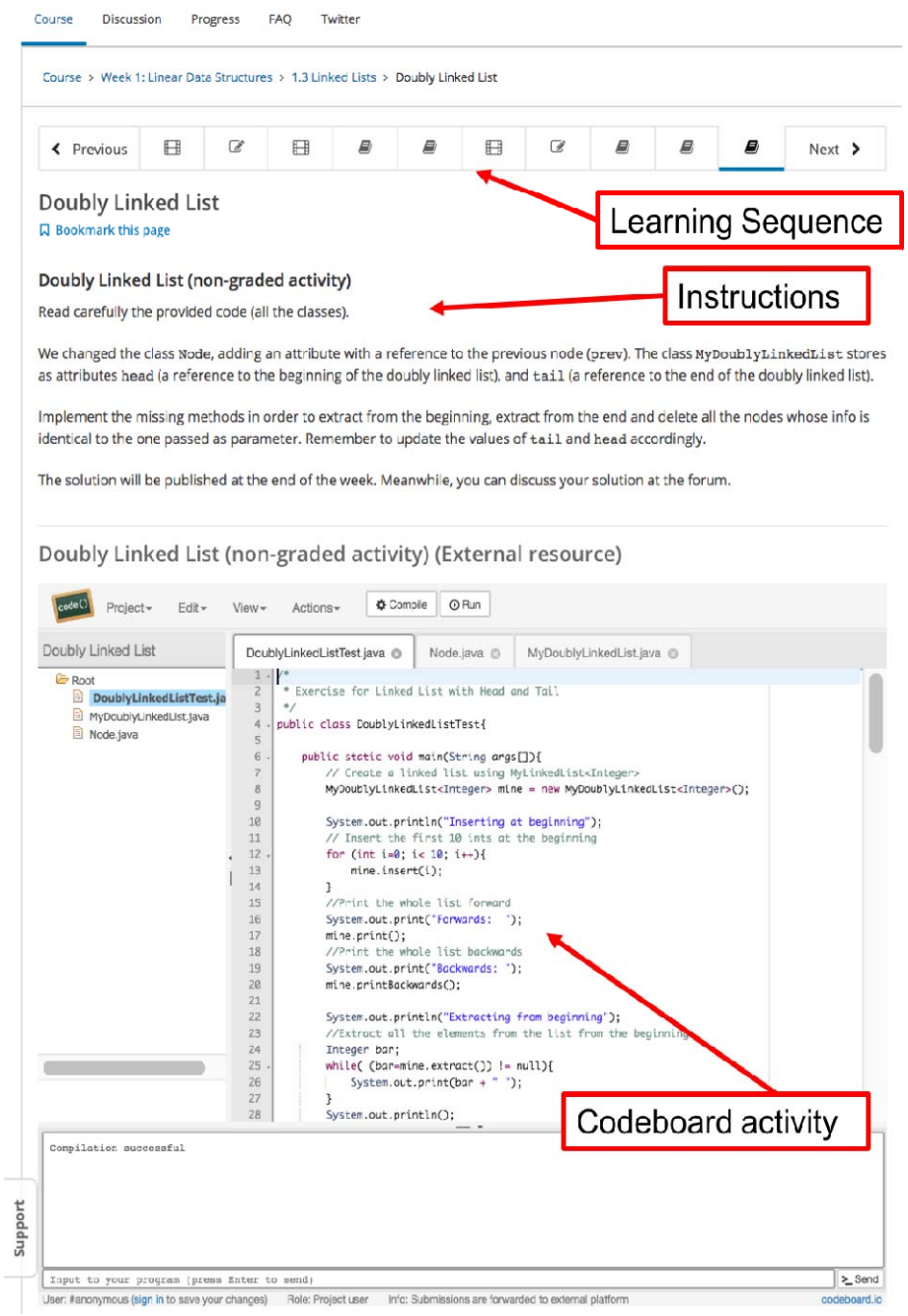

Figure 1: Example of integration of Codeboard (bottom) with a default Java code. This Codeboard activity is part of a learning sequence (top bar) of MOOC3. The text on top gives instructions on how to solve the activity.

It is important to note that it was not possible to match registered edX users with registered Codeboard users, as registration processes were completely independent from each other. Actually, in order to encourage access to Codeboard from edX, unregistered access needed to be enabled (i.e., access to Codeboard by unregistered, anonymous users).

4 Results and discussion

The data collected in this study aims to answer the three research questions presented in the introduction of this paper. These three research questions are particularized for the case of Codeboard, and the three MOOCs on "Introduction to Programming with Java" deployed in edX. It is important to note that although there was no asynchronous run for the third MOOC (which would have allowed comparison with the equivalent synchronous run), this third MOOC is included in the analysis as it contributes to answer RQ1 (collecting learners' opinions), and also to provide results on the number of accesses, compilations and runs, as part of RQ2 and RQ3.

4.1 (RQ1) Usefulness of Codeboard

1222 learners (about one fourth of total completers) filled in the final questionnaire in which they were asked about Codeboard (N=1222). The two questions related to Codeboard were formulated with a Likert scale between 1 and 5 (see Table 1). The usefulness of Codeboard activities was assessed with 4.12 on average (std.=1.02); it is noteworthy that 92.88% of learners assessed the usefulness as 3 or above. The difficulty of the Codeboard activities was assessed with 3.41 on average (std.=0.99) which indicates that for most learners the difficulty of the activities was not a problem. These results show that, from learners' perspective, Codeboard was a useful tool to promote learning-by-doing and did not disrupt learners' interaction and engagement in the MOOC.

Value	Usefulness of Codeboard activities	Difficulty of Codeboard activities
1	*36 (2.95%)*	*48 (3.93%)*
2	*51 (4.17%)*	*122 (9.98%)*
3	*203 (16.61%)*	*511 (41.82%)*
4	*376 (30.77%)*	*359 (29.38%)*
5	*556 (45.5%)*	*183 (14.98%)*
TOTAL	*1222 (100%)*	

Table 1: Learners' answers to survey questions on the usefulness of Codeboard activities (from 1 – useless, to 5 – very useful), and on the difficulty of Codeboard activities (from 1 – very easy to 5 – very difficult).

4.2 (RQ2) Learners' interactions in synchronous and asynchronous MOOCs

In order to evaluate learners' interactions in the synchronous and asynchronous modes of MOOC delivery, an analysis is carried out from three different perspectives: (1) accesses to Codeboard activities, (2) compilations of Codeboard activities and (3) running of Codeboard activities. An access occurs each time the unit containing the

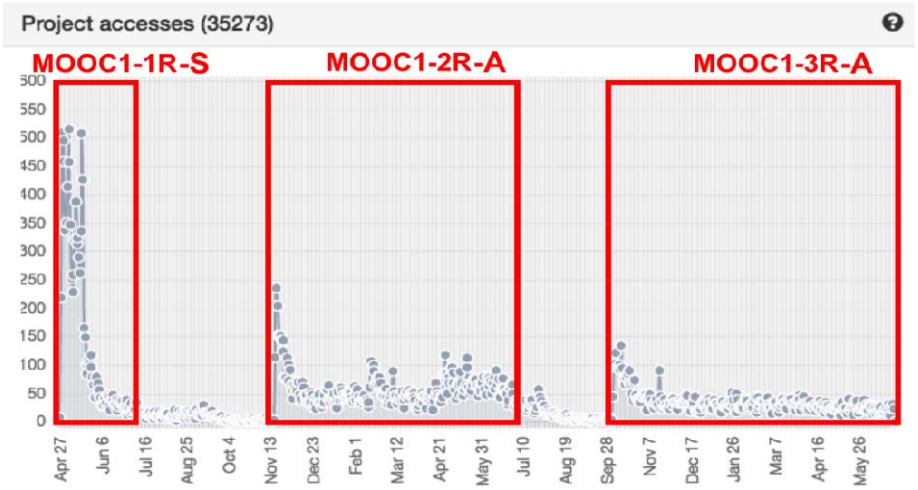

Figure 2: Number of daily accesses in an example Codeboard activity of MOOC1 in its different runs: first (MOOC1-1R-S, synchronous), second (MOOC1-2R-A, asynchronous) and third (MOOC1-3R-A, asynchronous). The total number of accesses in the timespan is 35,273.

Codeboard activity in the learning sequence is loaded. In addition, in order to be able to run a Java code, even with the default code, learners must first compile it. Moreover, every modification in the code requires compiling before running.

Figure 2 shows an example with the daily number of accesses for one of the Codeboard activities in MOOC1. The three runs are clearly marked in the figure: the first synchronous run (MOOC1-1R-S) and the following two asynchronous runs (MOOC1-2R-A and MOOC1-3R-A). It is worth considering two aspects regarding this figure. First, once a MOOC is closed it becomes archived and, although no more new enrolments are allowed, learners who were already enrolled can still access the course materials; this explains the number of residual accesses to this Codeboard activity in the period between runs. Second, it is worth mentioning that the first run was announced six months in advance before the starting date, while the following two runs were announced from two to three months in advance. As a consequence, the day the course started there were many more learners registered in the first run as compared to the second and third runs. Nevertheless, as these two runs are open for longer periods of time, they end up largely increasing the number of enrollments, with the second run even getting more enrollees than the first one at the end.

Figures 3 and 4 show the total number of accesses for each Codeboard activity of the first two MOOCs in their different runs; the third MOOC is not shown here as there are no data to compare the synchronous and asynchronous delivery modes. Figure 3 presents the results of the three runs of MOOC1, while Figure 4 presents the results of the two runs of MOOC2. These figures include both the activities that learners must solve and their reference solutions (indicated in parenthesis and with red background). Codeboard activities are arranged on the x-axis in the sequential order in which they appear in the MOOC. The lines representing the total number of

accesses per Codeboard activity follow the same pattern, both in synchronous and asynchronous modes, typically weighted by the number of registered learners. This is confirmed by the existing very strong, significant correlation between runs presented in Table 2 (second column).

	Number of accesses per Codeboard activity	Number of compilations per Codeboard activity	Number of runs per Codeboard activity
MOOC1-1R-S & MOOC1-2R-A	0.981	0.979	0.986
MOOC1-1R-S & MOOC1-3R-A	0.98	0.95	0.96
MOOC1-2R-A & MOOC1-3R-A	0.996	0.975	0.977
MOOC2-1R-S & MOOC2-2R-A	0.974	0.992	0.978

Table 2: Pearson correlation coefficient (ρ) between runs of MOOC1 and between runs of MOOC2 for the total number of accesses, compilations and runs of each Codebard activity ($p<0.01$).

Table 3 shows the average number of accesses per Codeboard activity in the three MOOCs. In MOOC1, the average number of accesses is consistent with the number of enrollees per run (93,574, 135,468 and 66,788). In MOOC2, the average number of accesses is higher in the first edition despite having obtained some fewer enrollees (13,630 and 14,997). MOOC3 has the least average number of accesses, and also enrollees (8,692), although it has no other runs to compare with. The three MOOCs has large SD, as can be seen in detail in Figures 3 and 4 for MOOC1 and MOOC2.

MOOC	1R-S (First run)	2R-A (Second run)	3R-A (Third run)
MOOC1	5173 (SD=3992)	6251 (SD=5369)	4198 (SD=3362)
MOOC2	993 (SD=455)	705 (SD=370)	-
MOOC3	267 (SD=233)	-	-

Table 3: Average number of accesses per Codeboard activity for the three MOOCs in their different runs.

In terms of number of compilations and runs of Codeboard activities, Table 2 (third and fourth columns) shows that once again there is a very strong, significant correlation between the different runs of MOOC1 and between the different runs of MOOC2. The lines representing the total number of compilations and the total number of runs per Codeboard activity follow once again the same pattern when comparing each MOOC, both in synchronous and asynchronous modes, and are typically weighted by the number of enrollees.

In order to better understand the meaning of these results, the number of compilations between the number of accesses, and the number of compilations between the number of runs were analyzed for each Codeboard activity. Figures 5 and 6 show the number of compilations divided by the number of accesses for each Codeboard activity in MOOC1 and MOOC2; Table 4 presents the average values for the three MOOCs in their different runs. Figures 7 and 8 show the number of compilations divided by the number of runs for each Codeboard activity in MOOC1 and MOOC2; Table 5 presents the average values for the three MOOCs in their different runs. In each figure, all lines follow a similar pattern regardless of whether the MOOC is offered in a synchronous or asynchronous mode.

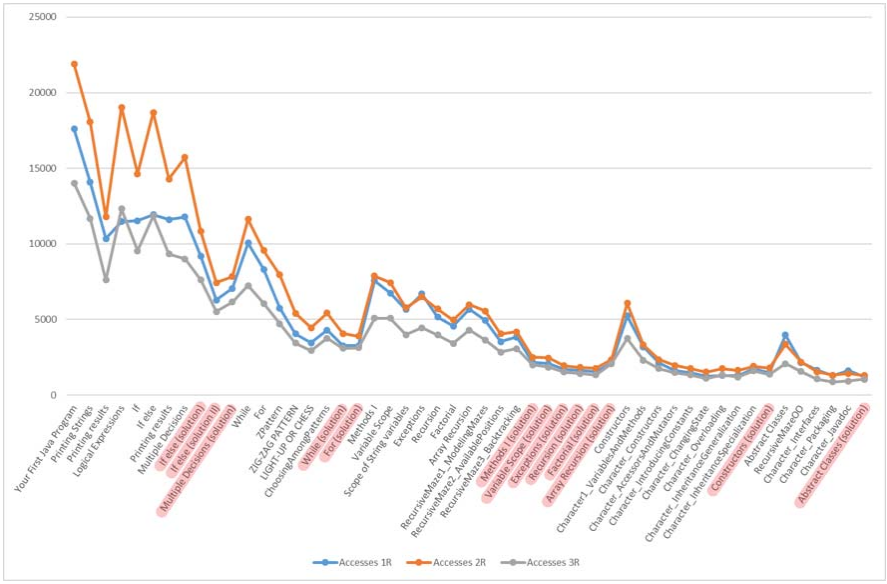

Figure 3: Number of accesses for each Codeboard activity in MOOC1 in its different runs: MOOC1-1R-S (blue), MOOC1-2R-A (orange), MOOC1-3R-A (grey). The figure includes the 38 Codeboard activities designed for MOOC1, plus the 13 solutions given to the most complex activities (with red background).

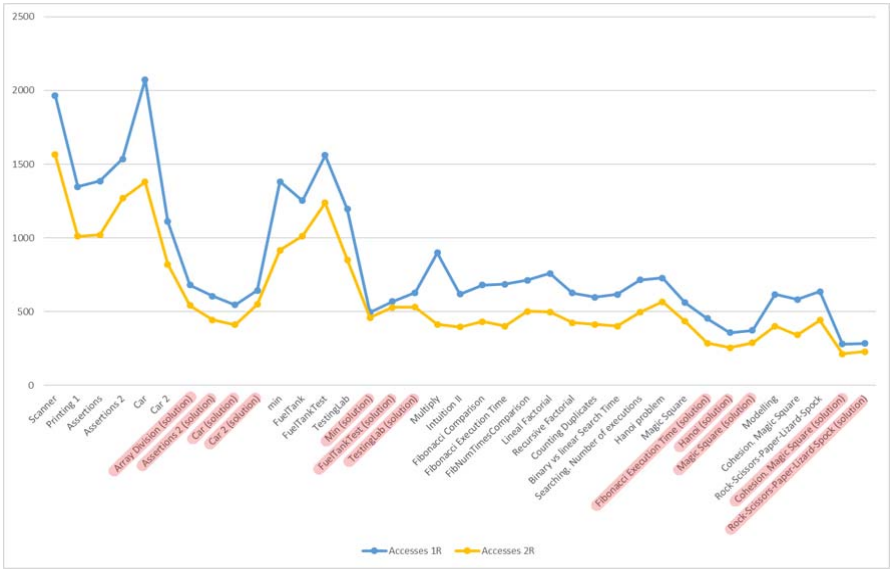

Figure 4: Number of accesses for each Codeboard activity in MOOC2 in its different runs: MOOC2-1R-S (blue), MOOC2-2R-A (orange). The figure includes the 21 Codeboard activities designed for MOOC2, plus the 12 solutions given to the most complex activities (with red background).

The figures allow pointing out two interesting findings. First, solutions to Codeboard activities usually present a ratio of compilations between accesses below one (unlike those which are not solutions), which indicates that learners access the solutions but do not bother to compile and run the given reference codes. Contrasting this data with that presented in Table 1, according to which the level of Codeboard activities is not considered too difficult, the explanation could be that most of those who access the solution have managed to solve the activity and simply observe the solution provided by the teachers. Second, the ratio between compilations and runs of code is usually between 1 and 2, but there are three large peaks in Figures 7 and 8. The default code provided by teachers always compiles, except in the three mentioned peaks (this was done on purpose). In these three activities, learners cannot run the code until they solve some existing compilation problems, which leads to the aforementioned higher ratio. In the rest of the activities, a ratio close to one indicates that either the learners do not have too many erroneous compilations, or, if so, they run the code several times to better understand how it works.

MOOC	1R-S (First run)	2R-A (Second run)	3R-A (Third run)
MOOC1	1.38 (SD=0.81)	1.5 (SD=0.88)	1.57 (SD=1.09)
MOOC2	1.09 (SD=0.66)	1.09 (SD=0.68)	-
MOOC3	0.53 (SD=0.51)	-	-

Table 4: Average of the number of compilations divided by the number of accesses per Codeboard activity for the three MOOCs in their different runs.

MOOC	1R-S (First run)	2R-A (Second run)	3R-A (Third run)
MOOC1	1.62 (SD=1.01)	1.51 (SD=0.7)	1.5 (SD=0.74)
MOOC2	1.69 (SD=1.71)	1.74 (SD=1.66)	-
MOOC3	1.42 (SD=0.59)	-	-

Table 5: Average of the number of compilations divided by the number of runs per Codeboard activity for the three MOOCs in their different runs.

4.3 (RQ3) Learners' engagement in synchronous and asynchronous MOOCs

To determine learners' engagement, the number of accesses to Codeboard activities by registered users are analyzed, as opposed to the total number of accesses (from both registered and anonymous users). Registered users can save their projects and continue working on them later. A user who does not register in Codeboard could be seen as less engaged with the MOOC as this learner does not really care about losing all the work done in Codeboard when moving forward in the learning sequence. In addition, there are complex activities that may take some time to be solved, so it is strongly recommended to save them in order to keep working on them in different moments.

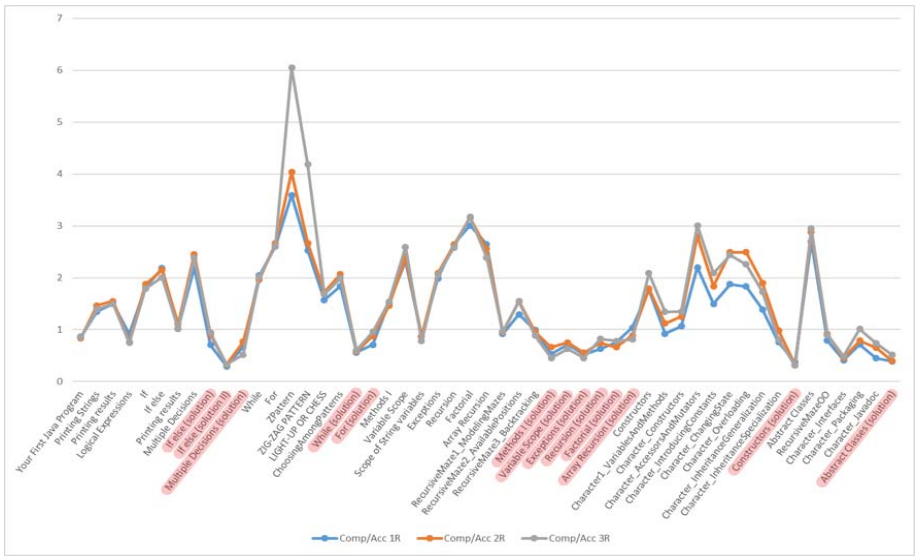

Figure 5: Number of compilations divided by number of accesses for each Codeboard activity in MOOC1 in its different runs: MOOC1-1R-S (blue), MOOC1-2R-A (orange), MOOC1-3R-A (grey). The figure includes the 38 Codeboard activities designed for MOOC1, plus the 13 solutions given to the most complex activities (with red background).

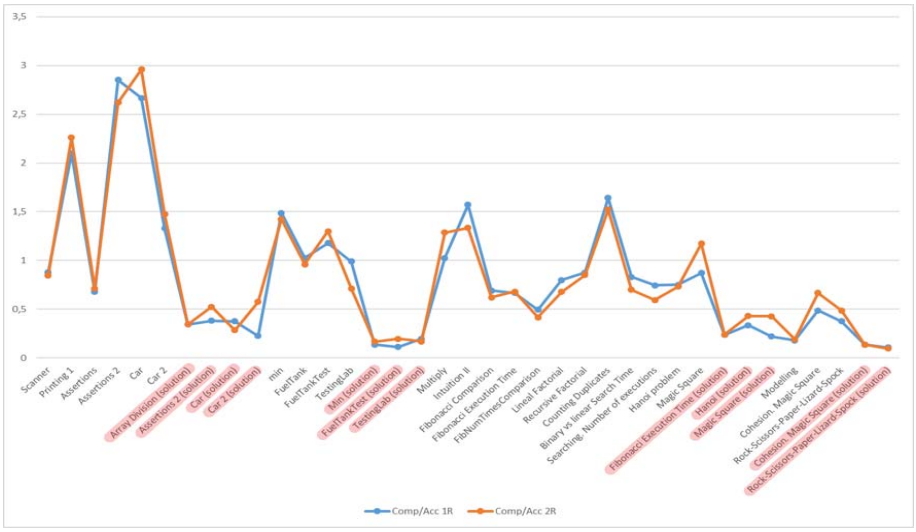

Figure 6: Number of compilations divided by number of accesses for each Codeboard activity in MOOC2 in its different runs: MOOC2-1R-S (blue), MOOC2-2R-A (orange). The figure includes the 21 Codeboard activities designed for MOOC2, plus the 12 solutions given to the most complex activities (with red background).

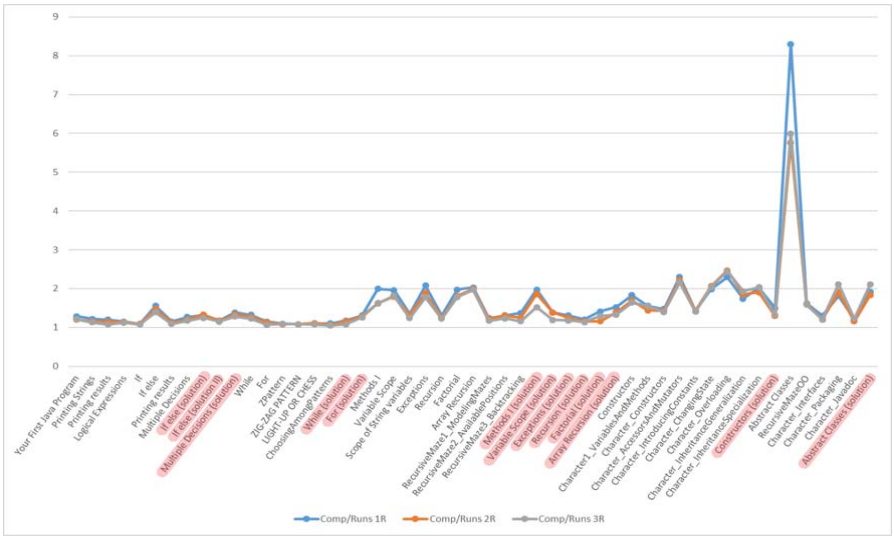

Figure 7: Number of compilations divided by number of runs for each Codeboard activity in MOOC1 in its different runs: MOOC1-1R-S (blue), MOOC1-2R-A (orange), MOOC1-3R-A (grey). The figure includes the 38 Codeboard activities designed for MOOC1, plus the 13 solutions given to the most complex activities (with red background).

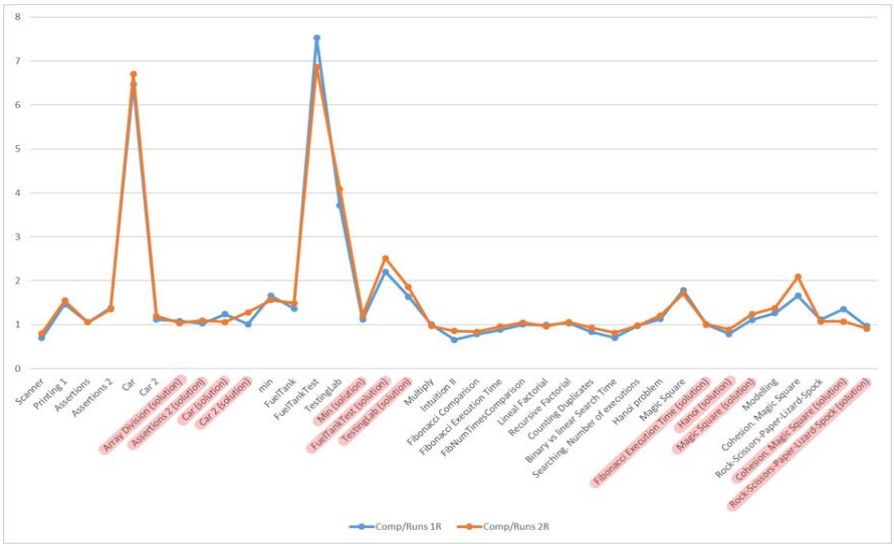

Figure 8: Number of compilations divided by number of runs for each Codeboard activity in MOOC2 in its different runs: MOOC2-1R-S (blue), MOOC2-2R-A (orange). The figure includes the 21 Codeboard activities designed for MOOC2, plus the 12 solutions given to the most complex activities (with red background).

Figures 9 and 10 show the number of accesses from registered learners divided by total number of accesses for each Codeboard activity in MOOC1 and MOOC2; Table 6 complements these figures, presenting the average values for the three MOOCs in their different runs. In this case, important differences can be observed between the synchronous and asynchronous modes. Learners are more engaged in the synchronous runs, both when accessing the activities to be solved and when accessing their solutions. In contrast to the very strong correlations obtained previously regarding the number of accesses, compilations and runs, weak, non-significant correlations are obtained when comparing the synchronous run of MOOC1 with each of the asynchronous runs. This correlation turns out to be moderate and significant (ρ=0.571, p-value<0.01) when comparing the two runs of the second MOOC, and very strong and significant (ρ=0.898, p-value<0.01), when comparing the two asynchronous runs of the first MOOC. It is interesting to note how in the first Codeboard activity of MOOC1 there are few accesses by registered learners compared to the total number of accesses. This first activity was very simple; it only shows a basic Java program and does not require any changes in the default code. The following Codeboard activities already required the learner to work on them, coding the missing parts, so it became more necessary to register and save the code.

	1R-S (First run)	2R-A (Second run)	3R-A (Third run)
MOOC1	0.337 (SD=0.04)	0.271 (SD=0.03)	0.266 (SD=0.04)
MOOC2	0.308 (SD=0.04)	0.213 (SD=0.03)	-
MOOC3	0.283 (SD=0.06)	-	-

Table 6: Average of the number of accesses from registered users divided by the total number of accesses per Codeboard activity

5 Limitations

This study presents several limitations. First, the amount of data available on the use of Codeboard is limited, as only high-level data obtained through its graphical user interface could be obtained. We are currently working on the installation of a local Codeboard instance at UC3M in order to collect low-level data with the aim to improve this research and reinforce the conclusions drawn. Second, the three MOOCs used for the analysis are independent in terms of syllabuses and certifications, but there is certain relationship among them as they form a series of courses. Therefore, there might be students who enrolled in several of them and, as a consequence, the data obtained from the MOOCs and their runs are not completely independent. A future line of work is to repeat this analysis with completely independent MOOCs, from different knowledge areas, and deployed in different platforms. The problem then is that it would be more difficult to find a third-party external tool which fits in the context of all the MOOCs under analysis. Third, the study analyzed the interaction of learners with Codeboard activities, but some learners may have taken the reference code provided by teachers to their usual development environment, outside Codeboard; so, it would be interesting to ask the learners where they actually worked on the proposed tasks, to see the number of them who choose Codeboard versus other

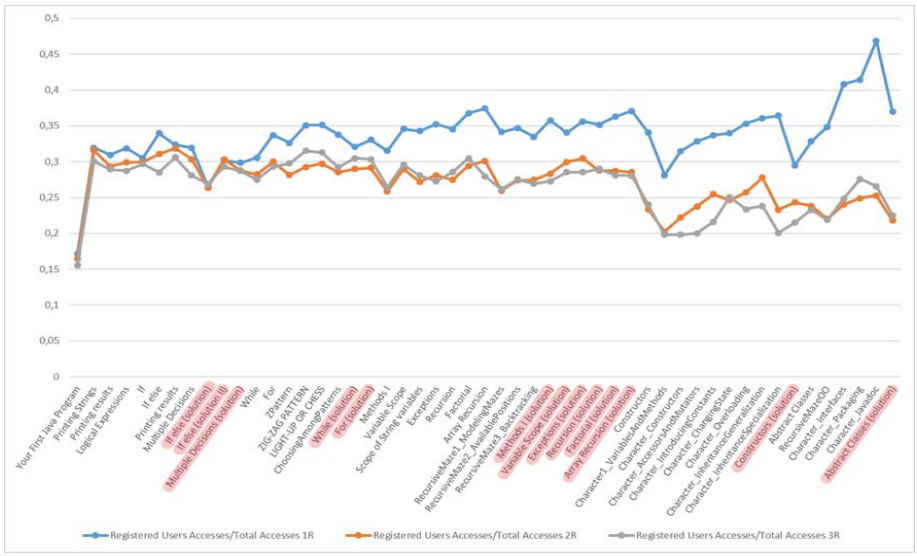

Figure 9: Number of accesses from registered learners divided by the total number of accesses for each Codeboard activity in MOOC1 in its different runs: MOOC1-1R-S (blue), MOOC1-2R-A (orange), MOOC1-3R-A (grey). The figure includes the 38 Codeboard activities designed for MOOC1, plus the 13 solutions given to the most complex activities (with red background).

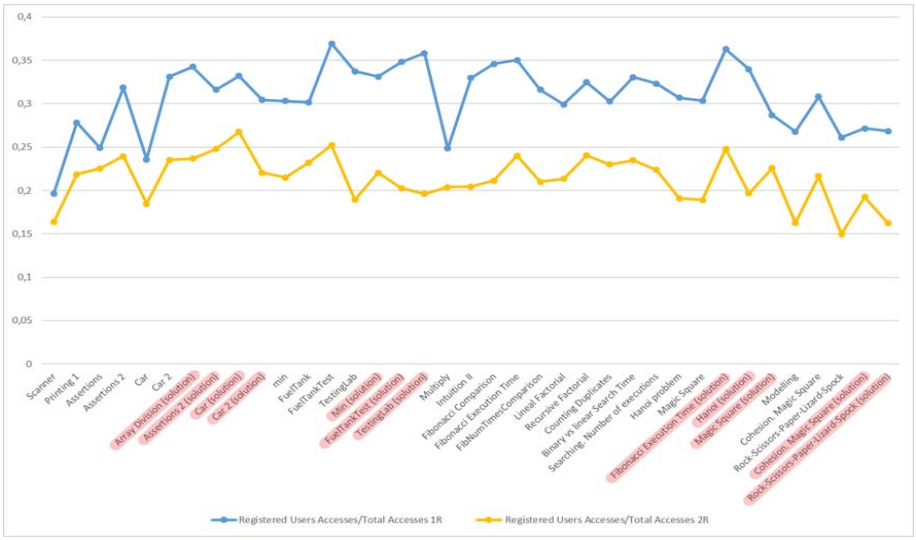

Figure 10: Number of accesses from registered learners divided by the total number of accesses for each Codeboard activity in MOOC2 in its different runs: MOOC2-1R-S (blue), MOOC2-2R-A (orange). The figure includes the 21 Codeboard activities designed for MOOC2, plus the 12 solutions given to the most complex activities (with red background).

locally installed development environments. In addition, the measure of the engagement could be calculated including also tools for code development that the learners install for taking these MOOCs. Last but not least, all the Codeboard activities proposed in these MOOCs were formative; so, it would be interesting to see how these results are affected by the fact that the proposed activities count towards the final grade.

6 Conclusions

MOOCs should be designed to promote learners' interaction and learning-by-doing, and not only as repositories of video lectures which replicate passive instruction models. This interaction can be promoted through the use of the built-in tools included in the MOOC platform, but also through the integration of third-party external tools, which are specific for the area of knowledge addressed by MOOC. The instructor knows the MOOC syllabus, and therefore what types of external tools may be appropriate to support the learning sequences in the MOOC. This paper has presented a study of the integration of Codeboard, a third-party external tool aimed at facilitating code development directly from the browser, in three different MOOCs on "Introduction to Programming with Java," offered in six runs throughout three years. Three of these runs (the first for each MOOC) were offered in a synchronous mode and the remaining three in an asynchronous running mode. This study allowed concluding that (1) Codeboard was a useful tool to promote learning-by-doing in the context of the three aforementioned MOOCs; (2) offering the MOOCs in synchronous or asynchronous running modes does not make a significant difference in terms of learners' interaction with Codeboard; (3) offering the MOOCs in a synchronous running mode results in greater engagement of learners with Codeboard, as compared to the asynchronous running mode.

Acknowledgements

The authors acknowledge the eMadrid Network, which is funded by the Madrid Regional Government (Comunidad de Madrid) with grant No. S2013/ICE-2715. This work also received partial support from the Spanish Ministry of Economy, Industry and Competitiveness, Project RESET (TIN2014-53199-C3-1-R), Project SYMBHYO-TIC (PTQ-15-07505), Project SIMLAP (RTC-2014-2811-1), Project SMARTLET (TIN2017-85179-C3-1-R), and from the European Commission through Erasmus+ projects MOOC-Maker (561533-EPP-1-2015-1-ESEPPKA2-CBHE-JP), SHEILA (562080-EPP-1-2015-1-BEEPPKA3-PI-FORWARD), COMPASS (2015-1-EL01-KA203-014033), and COMPETEN-SEA (574212-EPP-1-2016-1-NL-EPPKA2-CBHE-JP).

References

[Admiraal et al., 2015] Admiraal, W., Huisman, B., & Pilli, O.: "Assessment in Massive Open Online Courses," Electronic Journal of E-learning, 13, 4 (2015), 207-216.

[Alario-Hoyos and Wilson, 2010] Alario-Hoyos, C., & Wilson, S.: "Comparison of the main alternatives to the integration of external tools in different platforms," Proceedings of the International Conference of Education, Research and Innovation, ICERI 2010 (2010), 3466-3476.

[Alario-Hoyos et al., 2014] Alario-Hoyos, C., Pérez-Sanagustín, M., Delgado-Kloos, C., Parada G., H. A., & Muñoz-Organero, M.: "Delving into participants' profiles and use of social tools in MOOCs," IEEE Transactions on Learning Technologies, 7, 3 (2014), 260-266.

[Alario-Hoyos et al., 2016] Alario-Hoyos, C., Delgado Kloos, C., Estévez-Ayres, I., Fernández-Panadero, C., Blasco, J., Pastrana, S., Suárez-Tangil, G., & Villena-Román, J.: "Interactive activities: the key to learning programming with MOOCs," Proceedings of the Fourth European MOOCs Stakeholders Summit, EMOOCs 2016, (2016), 319-328.

[Aleven et al., 2016] Aleven, V., Baker, R., Wang, Y., Sewall, J., & Popescu, O.: "Bringing Non-Programmer Authoring of Intelligent Tutors to MOOCs," Proceedings of the Third ACM Conference on Learning@ Scale (2016), 313-316. ACM.

[Anido et al., 2001] Anido, L., Llamas M., & Fernández, M. J.: "Internet-based learning by doing," IEEE Transactions on Education, 44, 2 (2001), 193-201.

[Bali, 2014] Bali, M.: "MOOC pedagogy: gleaning good practice from existing MOOCs," Journal of Online Learning and Teaching, 10, 1 (2014), 44-56.

[Bhatnagar et al., 2016] Bhatnagar, S., Lasry, N., Desmarais, M., & Charles, E.: "DALITE: Asynchronous Peer Instruction for MOOCs," Proceedings of the European Conference on Technology Enhanced Learning, EC-TEL 2016 (2016), 505-508. Springer International Publishing.

[Breslow et al., 2013] Breslow, L., Pritchard, D. E., DeBoer, J., Stump, G. S., Ho, A. D., & Seaton, D. T.: "Studying learning in the worldwide classroom: Research into edX's first MOOC," Research & Practice in Assessment, 8 (2013), 13-25.

[Campbell et al., 2014] Campbell, J., Gibbs, A. L., Najafi, H., & Severinski, C.: "A comparison of learner intent and behaviour in live and archived MOOCs," The International Review of Research in Open and Distributed Learning, 15, 5 (2014), 235-262.

[Champaign et al., 2014] Champaign, J., Colvin, K. F., Liu, A., Fredericks, C., Seaton, D., & Pritchard, D. E.: "Correlating skill and improvement in 2 MOOCs with a student's time on tasks," Proceedings of the first ACM conference on Learning@ scale conference (2014), 11-20. ACM.

[Clarke and Braun, 2013] Clarke, V., & Braun, V.: "Teaching thematic analysis: Overcoming challenges and developing strategies for effective learning," The psychologist, 26, 2 (2013), 120-123.

[Crosslin, 2018] Crosslin, M. "Exploring self-regulated learning choices in a customisable learning pathway MOOC," Australasian Journal of Educational Technology, 34, 1 (2018), 131-144.

[Cruz-Benito et al., 2015] Cruz-Benito, J., Borrás-Gené, O., García-Peñalvo, F. J., Blanco, Á. F., & Therón, R.: "Extending MOOC ecosystems using web services and software architectures," Proceedings of the XVI international conference on Human Computer Interaction (2015), 52. ACM.

[Delgado-Kloos et al., 2017] Delgado-Kloos, C., Alario-Hoyos, C., Estévez-Ayres, I., Muñoz-Merino, P. J., Ibáñez, M. B., & Crespo-García, R. M.: "Boosting interaction with educational

technology," Proceedings of the IEEE Global Engineering Education Conference, EDUCON 2017 (2017), 1763-1767, IEEE.

[Felder and Brent, 2003] Felder, R. M., & Rebecca B.: "Learning by doing," Chemical Engineering Education, 37, 4 (2003), 282-309.

[Fontela et al., 2011] Fontenla, J., Pérez, R., & Caeiro, M: "Using IMS Basic LTI to integrate games in LMSs: Game•Tel," Proceedings of the IEEE Global Engineering Education Conference, EDUCON 2011 (2011), 299-306. IEEE.

[Forment et al., 2012] Forment, M. A., Guerrero, M. J. C., Mayol, E., Piguillem, J., Galanis, N., García-Peñalvo, F. J., & González, M. Á. C.: "Docs4Learning: Getting Google Docs to work within the LMS with IMS BLTI," Journla of Universal Computer Science, 18, 11 (2012), 1483-1500.

[Hew, 2016] Hew, K. F.: "Promoting engagement in online courses: What strategies can we learn from three highly rated MOOCS," British Journal of Educational Technology 47, 2 (2016), 320-341.

[IMS, 2012] IMS Global Learning Consortium: "IMS Global Learning Tools Interoperability Implementation Guide, Final Version 1.1" (2012). Published online: http://www.imsglobal.org/specs/ltiv1p1/implementation-guide

[Krugel and Hubwieses, 2017] Krugel, J., & Hubwieser, P.: "Computational thinking as springboard for learning object-oriented programming in an interactive MOOC," Proceedings of the IEEE Global Engineering Education Conference, EDUCON 2017 (2017) 1709-1712.

[Morales Chan et al., 2017] Morales Chan, M., de La Roca, M., Alario-Hoyos, C., Barchino Plata, R., Medina, J.A., & Hernández Rizzardini, R.: "Perceived usefulness and motivation students towards the use of a cloud-based tool to support the learning process in a Java MOOC," Proceedings of the International Conference MOOC-Maker 2017 (2017), 73-82.

[Queirós et al., 2016] Queirós, R., Leal, J. P., & Paiva, J. C.: "Integrating rich learning applications in LMS," In Li, Chang, Kravcik, Popescu, Huang, Kinshuk, Chen (Eds.) State-of-the-art and future directions of smart learning, Springer Singapore, (2016), pp. 381-386.

[Rai and Chun-Rao, 2016] Rai, L., & Chun-Rao, D.: "Influencing factors of success and failure in MOOC and general analysis of learner behavior," International Journal of Information and Education Technology, 6, 4 (2016), 262-268.

[Rizzuto, 2017] Rizzuto, M.: "Design Recommendations for Self-Paced Online Faculty Development Courses," TechTrends, 61, 1 (2017), 77-86.

[Staubitz et al., 2014] Staubitz, T., Renz, J., Willems, C., Jasper, J., & Meinel, C.: "Lightweight ad hoc assessment of practical programming skills at scale," Proceedings of the IEEE Global Engineering Education Conference, EDUCON 2014 (2014), 475-483, IEEE.

[Staubitz and Meinel, 2017] Staubitz, T., & Meinel, C.: "Collaboration and Teamwork on a MOOC Platform: A Toolset," Proceedings of the Fourth ACM Conference on Learning@ Scale (2016), 165-168).

[Suen, 2014] Suen, H. K.: "Peer assessment for massive open online courses (MOOCs)," The International Review of Research in Open and Distributed Learning, 15, 3 (2014), 312-327.

[Watson, et al. 2017] Watson, S. L., Watson, W. R., Yu, J. H., Alamri, H., & Mueller, C.: "Learner profiles of attitudinal learning in a MOOC: An explanatory sequential mixed methods study," Computers & Education, 114 (2017), 274-285.